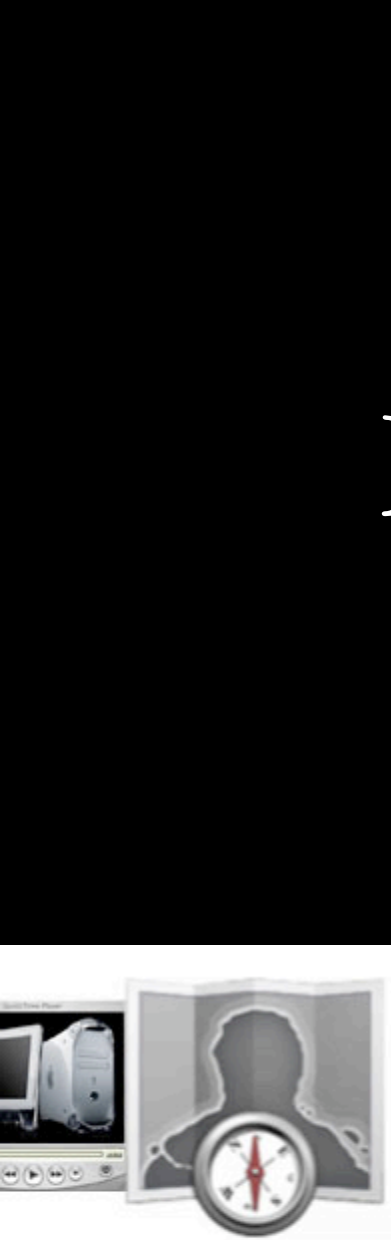


Improving Performance With Carbon Events

Session 207

Improving Performance With Carbon Events

Xavier Legros
Mac OS X Evangelist

Improving Performance With Carbon Events

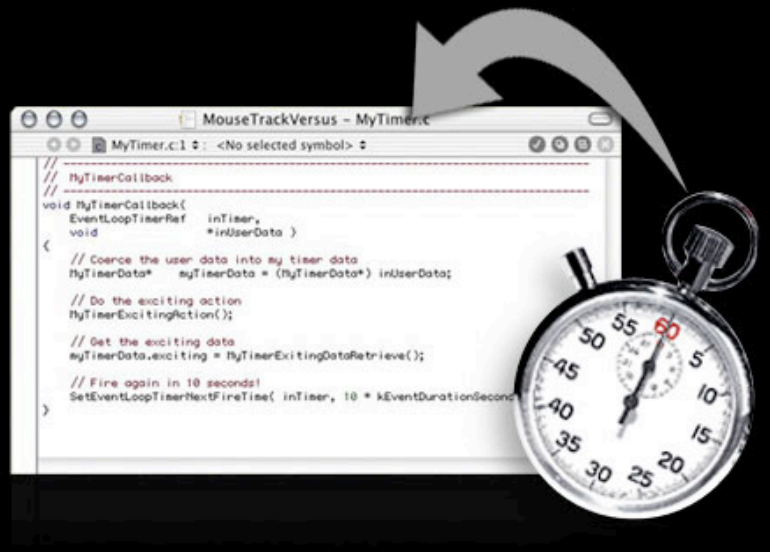
David McLeod
Appearance Slave
High Level Toolbox Team

Overview

- Timers
- Mouse Tracking
- Notifications
- Tracking Loops
- Stop calling FindWindow
- Stop calling EventAvail

What Is a Timer?

- A Carbon Event timer is a way to request delayed execution of code
- Controlled by the event loop
- Time based
- Code stays with implementation



Timers and Performance

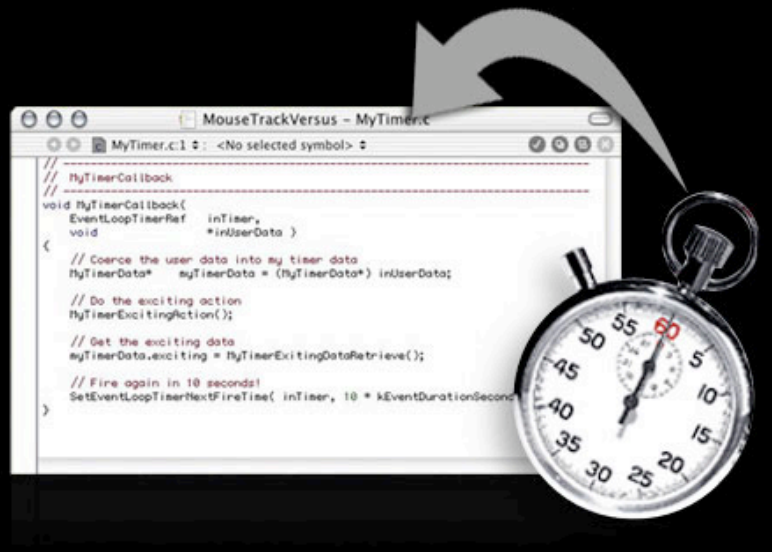
- No continuous checks to see if it is time
- No wasted time watching and processing null events
- Eliminates all timing overhead

Kinds of Timers

- One Shot Timer
- Periodic Timer
- Idle Timer
    - New in Jaguar!

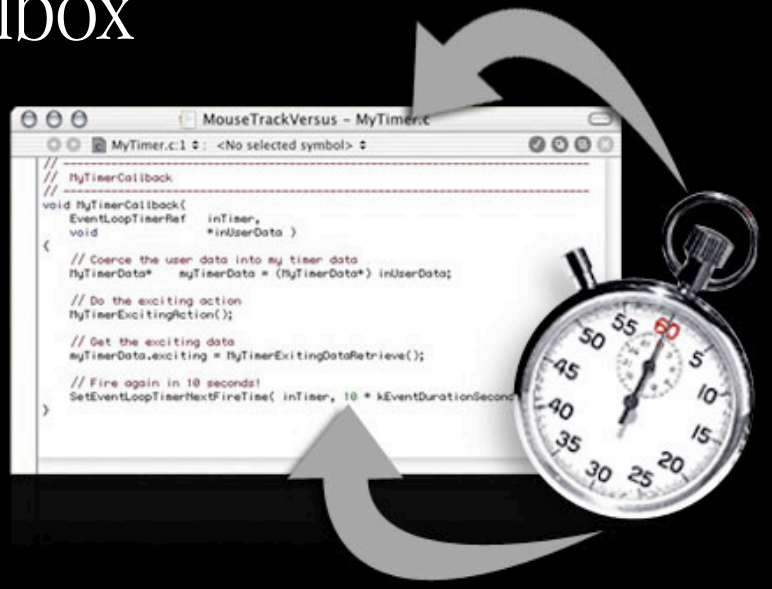
One Shot Timer

- Fires once
- Can be reset to re-fire
- Can be delayed or cancelled before execution
- Examples:
    - Dismiss a dialog after 2 minutes



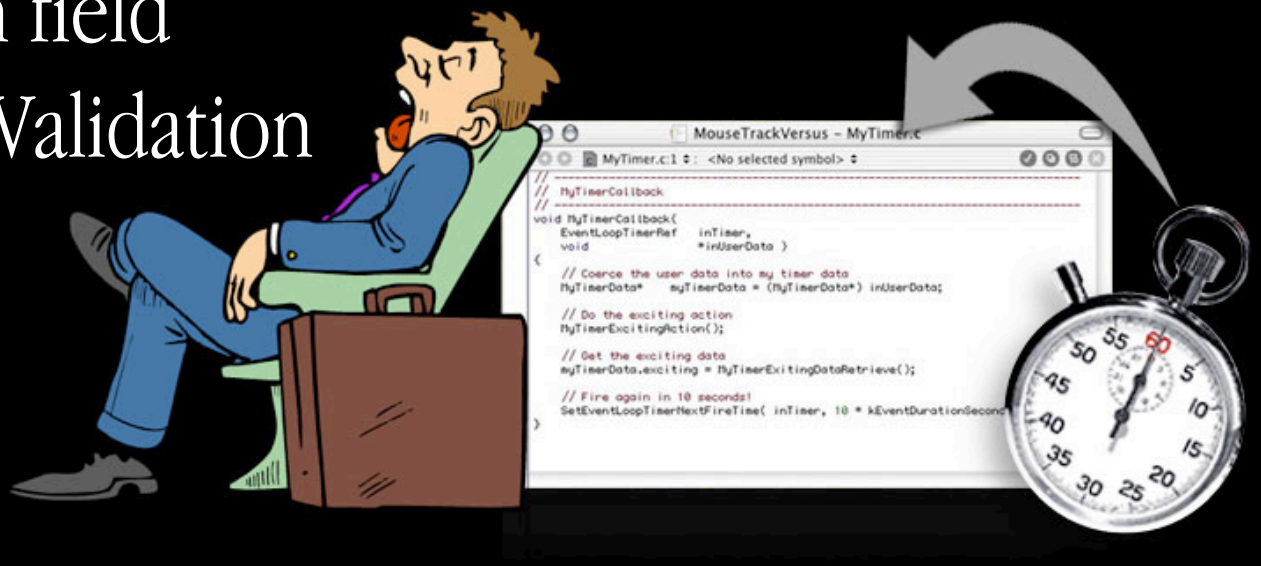
Periodic Timer

- Fires at specified intervals
- Can be delayed or cancelled before execution
- Examples:
    - Animation in the Toolbox



Idle Timer

- Fires when user becomes inactive
- Can be cancelled before execution
- Examples:
  - Search field
  - Input Validation



Making a One Shot Timer

```
err = InstallEventLoopTimer(
    GetCurrentEventLoop(),
    10 * kEventDurationSecond,   /* fire time */
    0 /* just fire once */,
    myTimerCallbackUPP,
    (void*) myUserData, &myTimerRef );
```

Making a Periodic Timer

```
err = InstallEventLoopTimer(
    GetCurrentEventLoop(),
    10 * kEventDurationSecond,   /* first fire */
    2 * kEventDurationSecond,    /* frequency */
    myTimerCallbackUPP,
    (void*) myUserData, &myTimerRef );
```

Making an Idle Timer

```
err = InstallIdleTimer(
        GetCurrentEventLoop(),
        10 * kEventDurationSecond,
        0 /* just fire once */,
        myIdleTimerCallbackUPP,
        (void*) myUserData, &myTimerRef );
```

More on Event Loop Timers

```
err = SetEventLoopTimerNextFireTime(
    myTimerRef,
    30 * kEventDurationMillisecond );

err = RemoveEventLoopTimer(
    myTimerRef );
```

Demo

Event Timer

Curt Rothert
Event Tamer
High Level Toolbox Team

Mouse Region Tracking

- Track the mouse to display status
  - Change the cursor for different areas
  - Display status for different areas
  - Change behavior for different areas

Old Way: Mouse Moved Events

- With 10.0, you could do this with a Mouse Moved event handler
- Too many mouse moved events
- Expensive

Use Tracking Regions

- New for Jaguar
- Register “hot” regions
- Callbacks track entry and exit of the regions
- Now we can ignore mouse moved events
- Can work on inactive windows!
- Performance gain!

When You Could Use Them

- Mouse leaves the window
- Cursor Setting
- Widget rollover
    - Change an image
    - Change a title
- Status tracking

Tracking Region How To

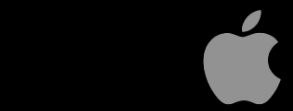
```
MouseTrackingRegionID  myRegionID =
        { kMyRegionSignature, kMyRegionID };

err = CreateMouseTrackingRegion(
        window, region, clip, options,
        myRegionID, userData, notifyTarget,
        &myTrackingRef );

/* Register for the events  */

/*
        continue on, notifyTarget will
        be notified
*/
```

Mouse Tracking Region Events

- kEventMouseEntered
- kEventMouseExited

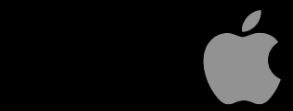
Other Mouse Tracking APIs

- ReleaseMouseTrackingRegion
- GetMouseTrackingRegionID
- ChangeMouseTrackingRegion
- MoveMouseTrackingRegion
- SetMouseTrackingRegionEnabled



Getting Region ID

```
err = GetEventParameter( inEvent,
    kEventParamMouseTrackingRef,
    typeMouseTrackingRef, NULL,
    sizeof( MouseTrackingRef ), NULL,
    &trackingRef );

err = GetMouseTrackingRegionID(
     trackRef, &regionID );

if ( regionID.signature == kMyAppSignature
     && regionID.id == kMyRegionID )
          /* do my region tracking stuff */
```



Constrained MouseMove

- If you really need MouseMoved information
- Use Tracking Regions to constrain your MouseMoved handler
- Install the MouseMoved handler on entry
- Remove the handler on exiting

Demo

Mouse Tracking

Curt Rothert
Combo Boxer
High Level Toolbox Team

Notifications

- Toolbox sends many notifications
- They are just Carbon Events that you can easily watch for

Notifications How To

- Just register for the Carbon Events!
  - kEventWindowHidden
  - kEventAppHidden
  - kEventMenuPopulate
  - kEventControlValueField-Changed
  - kEventVolumeMounted
  - Etc . . .

Notification Example

```
EventList     eventList = {
        { kEventClassVolume,
              kEventVolumeMounted },
        {  kEventClassVolume,
              kEventVolumeUnmounted }  };

err = InstallApplicationEventHandler(
        myVolumeMountingHandlerUPP,
        GetEventTypeCount( eventList ),
        eventList, NULL, NULL );
```

Tracking Loops

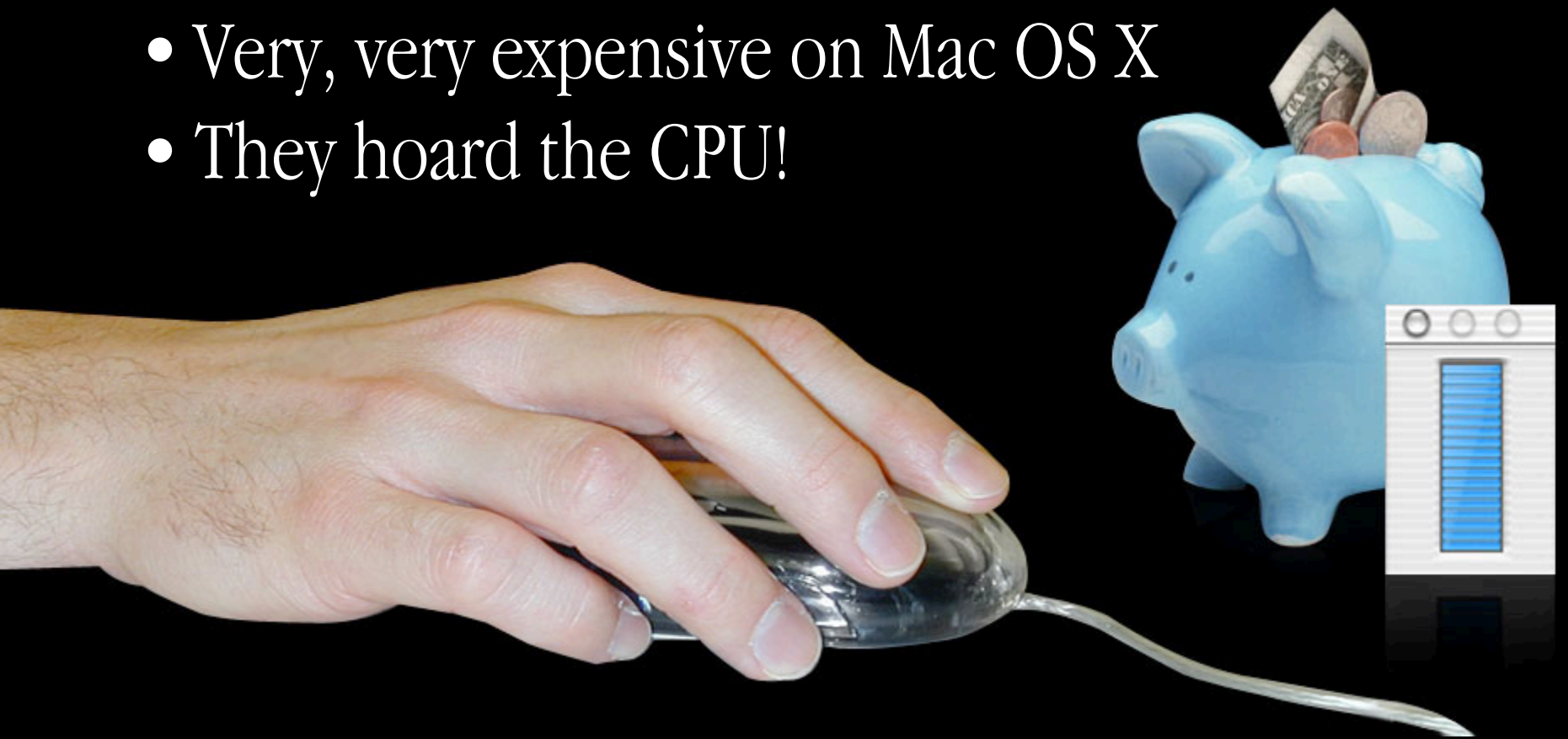
- Keeping track of mouse modifiers
- Keeping track of keyboard modifiers

CPU Hog

- Don’t call StillDown in tight loops
- Don’t call Button in tight loops
- Very, very expensive on Mac OS X
- They hoard the CPU!

CPU Hog

- Don’t call StillDown in tight loops
- Don’t call Button in tight loops
- Very, very expensive on Mac OS X
- They hoard the CPU!



Use Tracking Loop Instead!

- Call TrackMouseLocation or TrackMouseRegion
- No rainbow cursor!
- Timers fire!
- Frees CPU entirely

Use Tracking Loop Instead!

- Call TrackMouseLocation or TrackMouseRegion
- No rainbow cursor!
- Timers fire!
- Frees CPU entirely

Mouse Tracking How To

```
err = TrackMouseLocation( port,
        &outPt, &trackingResult );


switch ( trackingResult )
{
        case kMouseTrackingMouseDown:
        /* Do mouse down stuff */
        break;

        case kMouseTrackingMouseUp:
        /* Do mouse up stuff */
        break;

        etc…
}
```

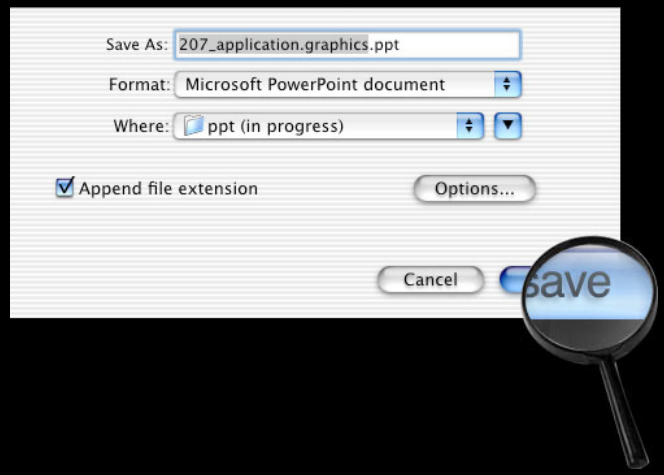
Demo

TrackMouseLocation

Curt Rothert
Combo Boxer
High Level Toolbox Team

FindWindow

- Developers used FindWindow to find where events occurred
- FindWindow has to call the Window Server and this is a performance hit
- All the info you need is in your Carbon Event!
- Avoid calling FindWindow



Carbon Event Parameters

- kEventMouseDown
  - kEventParamDirectObject (WindowRef)
  - kEventParamMouseLocation (Point)
  - kEventParamKeyModifiers (UInt32)
  - kEventParamWindowDefPart (WindowDefPartCode)
  - kEventParamControlRef (ControlRef)

Carbon Event Parameters

- kEventControlTrack
  - kEventParamDirectObject (ControlRef)
  - kEventParamMouseLocation (Point)
  - kEventParamKeyModifiers (UInt32)
  - kEventParamControlAction (ControlActionUPP)
  - kEventParamControlPart (ControlPartCode)

EventAvail: Inefficient

- Developers use EventAvail to determine the state of the keyboard, the mouse, etc . . .
- EventAvail trolls the event queue and runs the Event Loop
- Queue is thread safe
  - Synchronization cost
- Running Event Loop flushes windows and fires timers
- Avoid calling EventAvail

Check Loop: Much Better

- Don’t tickle the event queue to find out where the mouse is or the state of the keyboard
- GetGlobalMouse
    - Calls Window Server
- GetCurrentKeyModifiers
    - Calls Window Server

Even Better Than That! (Jaguar)

- GetCurrentEventButtonState
    - No call to Window Server
- GetCurrentEventKeyModifiers
    - No Call to Window Server

Checking the Event Queue

- Check queue for user cancel
- CheckEventQueueForUserCancel
- How descriptive!
- Doesn't call the event loop
- Toolbox can check this faster than checking everything

Ultimate: Don’t Check At All

- You might already have the info, so why check?
- Carbon Events often include modifiers
- Carbon Events often include mouse location

Carbon Event Parameters

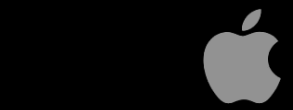
- kEventControlTrack
  - kEventParamDirectObject (ControlRef)
  - kEventParamMouseLocation (Point)
  - kEventParamKeyModifiers (UInt32)
  - kEventParamControlAction (ControlActionUPP)
  - kEventParamControlPart (ControlPartCode)

Other New Carbon Events

- kEventAppFocusMenuBar
- kEventAppFocusNext-DocumentWindow
- kEventAppFocusNext-FloatingWindow
- kEventAppFocusToolbar
- kEventAppHidden
- kEventAppShown
- kEventControlDragEnter
- kEventControlDragLeave
- kEventControlDragReceive
- kEventControlDragWithin
- kEventControlEnabled-StateChanged
- kEventControlGetAuto-ToggleValue
- kEventControlGetClick-Activation

More New Carbon Events

- kEventControlGetNext-FocusCandidate
- kEventControlGetSize-Constraints
- kEventControlGet-SubviewForMouseEvent
- kEventControlHilite-Changed
- kEventControlTitle-Changed
- kEventMouseEntered
- kEventMouseExited
- kEventTextInputUnicode-Text
- kEventWindowCollapsing
- kEventWindowExpanding
- kEventWindowFocus-Content
- kEventWindowFocus-Toolbar
- kEventWindowGetDock-TileMenu



Additional Performance Tips

- Control Manager
- Menu Manager
- Some QuickDraw drawing performance

Control Manager

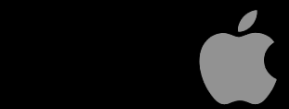
- Redrawing can be slow
- Modifying controls can force a redraw
- Really good example
    - Adding many items to a Popup Button control
- Use SetControlVisibility
    - Hide controls before changing many attributes
    - Show controls afterward

Speedy Control Changes

```
SetControlVisibility( popupControl,
        false, false /* Don't draw */ );

for ( i = 0; i < menuItems; i++ )
{
        /*
                Add items to your Popup control
        */
}

SetControlVisibility( popupControl,
        true, true /* Draw! */ );
```

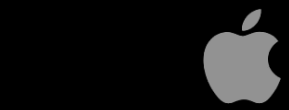
Menu Manager

- Don’t invalidate the menu key cache
    - Forces a rebuild
- Things that invalidate the cache
    - Changing a menu item modifiers or command key
    - Inserting and deleting menus
    - Inserting menu items with command keys
    - Clearing the menu bar or setting a new one
    - Changing submenus



Menu Manager

- How is my App invalidating the key cache?
    - Use gdb
    - Break on _InvalidateMenuKeyCache



Who Is the Cache Invalidator?

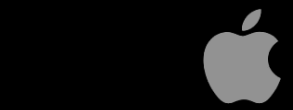
```
(gdb) break _InvalidateMenuKeyCache
Breakpoint 3 at 0x9acd06e8
(gdb) continue
Continuing.
Breakpoint 3, 0x9acd06e8 in
_InvalidateMenuKeyCache
(gdb) backtrace
#0  0x9acd06e8 in _InvalidateMenuKeyCache
#1  0x9acba605 in _SetItemCmd
#2  0x000029c4 in MyAppEventHandler
...
```

Menu Manager

- Use kEventUpdateCommandStatus and EnableItems wisely
    - Do not update everything!
    - Only enable or disable the requested menu
    - Listen to kEventParamMenuContext parameter

Menu Manager

- Use SetMenuItemData instead of making multiple calls to setter APIs
    - Mapping the MenuRef and MenuItemIndex to internal structure takes time



Slower Menu Item Setting

- Instead of:

```
SetItemCmd( menu, item, 'A' );
SetMenuItemModifiers( menu, item,
        kMenuShiftModifier );
SetMenuItemCommandID( menu, item,
        kHICommandReverseSelection );
```

Speedy Menu Item Setting

- Do this:

```
MenuItemDataRec    itemData;
itemData.whichData =
        kMenuItemDataCmdKey |
        kMenuItemDataCmdKeyModifiers |
        kMenuItemDataCommandID;

itemData.cmdKey = 'A';
itemData.cmdKeyModifiers = kMenuShiftModifier;
itemData.cmdID =
        kHICommandReverseSelection;

SetMenuItemData( menu, item, false,
        &itemData );
```

QuickDraw Tip 1

- QDFlushPortBuffer
  - Do not call it more often than refresh rate of the screen
  - Do not call it unnecessarily

QuickDraw Tip 2

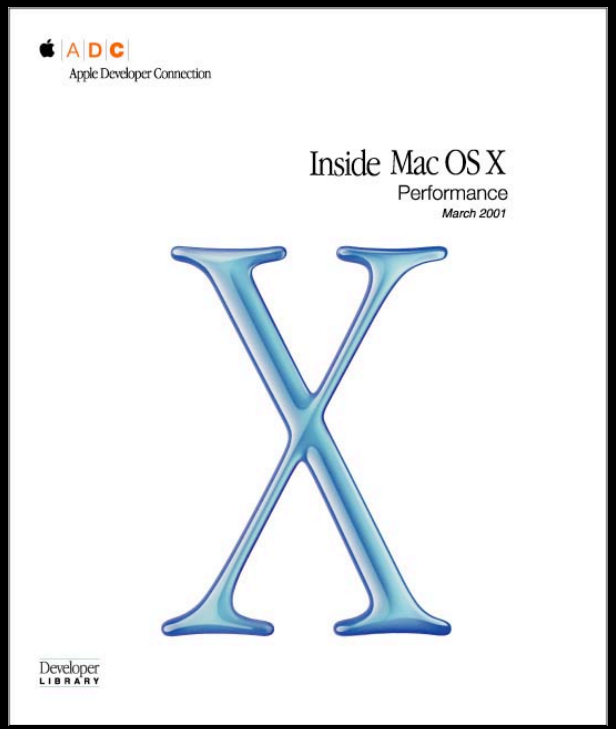
- Complex drawing with many QD calls
  - Before starting to draw, use QDAddRectToDirtyRegion to set the dirty region to a large, simple, rectangular region
  - Covers your drawing area
  - Dirty region calculations become simple
  - Don’t ignore the existing DirtyRgn!

QuickDraw Tip 3

- Before large number of drawing operations
  - Consider calling LockPortBits on your port
  - Only if drawing time is very brief!
    - < 1 or 2 seconds
  - If you leave the port locked too long, your window might have redraw issues
  - Don’t forget to UnlockPortBits!
- Session 516 on Friday: QuickDraw Performance

Documentation

Performance and the Carbon Event Manager



- Performance
- Handling Carbon Events
- Carbon Event Manager Reference

Documentation > Carbon > Operating System Services > Carbon Event Manager
Documentation > Documentation Essentials

Summary

- Timers
- Mouse Tracking
- Notifications
- Tracking Loops
- Stop calling FindWindow
- Stop calling EventAvail

Roadmap

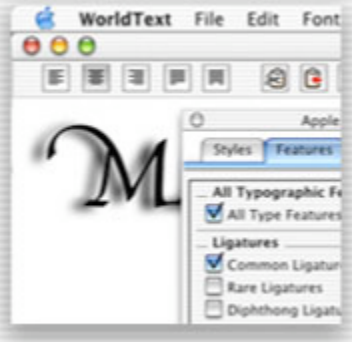
204 HIToolbox: An Architectural Overview	Hall 2 **Wed., 9:00am**
205 HIToolbox: Introducing HIView	Hall 2 **Wed., 10:30am**
206 HIToolbox: New Controls and Services	Hall 2 **Wed., 2:00pm**
203 Migrating to Carbon Events	Hall 2 **Tue., 5:00pm**

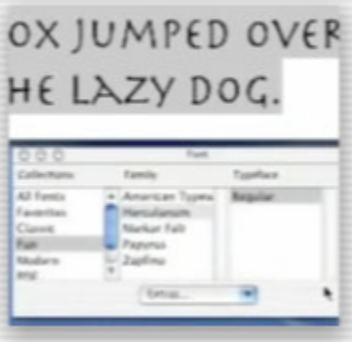
Who to Contact

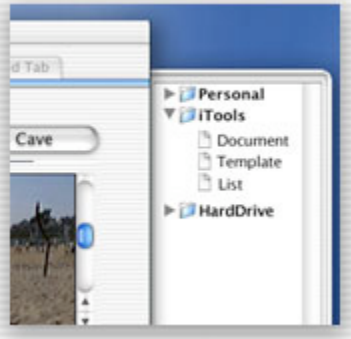
Xavier Legros
Mac OS X Evangelist
xavier@apple.com

http://developer.apple.com/wwdc2002/urls.html

Q&A







Xavier Legros
Mac OS X Evangelist
xavier@apple.com

http://developer.apple.com/wwdc2002/urls.html


WWDC2002


WWDC2002


WWDC2002